# IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of

| | |
|---|---|
| **JAMES** | Atty. Ref.: 36-1116 |
| Serial No.: 09/029,581 | Group Art Unit: 2171 |
| Filed: March 6, 1998 | Examiner: Al Hashemi, S. |
| For: DATABASE ACCESS | |

July 29, 2004

## THIRD APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Appellant hereby appeals the Rejection of April 29, 2004, Paper No. 26.

## REAL PARTY IN INTEREST

The real party in interest is the assignee, British Telecommunications plc, a corporation of the country of Great Britain.

## RELATED APPEALS AND INTERFERENCES

The Appellant, the undersigned, and the assignee are not aware of any related appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

## STATUS OF THE CLAIMS

Claims 1-9 remain pending in this application. Claims 1-9 stand rejected by the Examiner, the rejections of which are appealed. No claims have been substantively allowed.

## STATUS OF ANY AMENDMENT FILED SUBSEQUENT TO REJECTION

No Amendment has been filed subsequent to the Rejection of April 29, 2004. The claims as presented in the appendix to this brief are as amended by the March 6, 1998 Preliminary Amendment.

While no Amendment was filed subsequent to the Rejection of April 29, 2004, the undersigned and the Examiner's Supervisor conducted an interview on June 18, 2004. During the interview, it was agreed that the rejection of claims 1, 6 and 9 under 35 U.S.C. §101 would be withdrawn. (See the Interview Summary mailed June 21, 2004, paper no. 31.)

## CONCISE EXPLANATION OF THE INVENTION

The present invention relates to a client/server computer environment in which data consistency between data held in a master database and data held in a cache database is checked. In particular, the data consistency is checked by comparing a key associated with a respective cache database entry and a key associated with an index to a corresponding data entry in the master database. An exemplary embodiment of the present invention is shown in Figs. 1, 3c and 6. The structure shown in Figs. 3a and 3b and processes shown in Figs. 4 and 5 illustrate acknowledged prior art systems which can be read in contrast to Figs. 3c and 6 to understand the exemplary embodiment of the present invention.

The computer environment in accordance with an exemplary embodiment of the present invention includes, inter alia, a file server 100 which runs network and database management system (DBMS) software suitable for providing remote database access to a plurality of clients 130 over a network 140. The file server 100 includes a main memory 104 which is split into areas for standard program and data storage. The main memory 104 includes a large area of memory known as a system global area (SGA) 105 which can be used by the database software to speed up access to a master database 126 for the clients 130. (See Fig. 1 and page 7, lines 30-36).

The clients 130 are standard computer systems which run software suitable for reading and writing data stored in a database. The clients 130 each include a standard disk drive 135 having a storage area for a cache database 136 into which data from the master database 126 can be copied and accessed. (See Fig. 1 and page 8, lines 1-8).

The master database 126 is operatively coupled to the file server 100 and includes a data table 204 for storing a group of related data. The data table 204 is split into a plurality of pages 215 each comprising rows of data 210. The master database 126 also includes an index 206 comprising index pages 220. The index pages 220 contain index entries 225 for each row 210 in the data table 204. Each index entry 225 typically comprises index fields, one of which references a specific row of the data pages 215. (See Fig. 2 and page 8, lines 9-27).

The cache database 136 comprises a storage area 230 for holding cached data. The cached data typically comprises data rows 235 which are copied from the master database 126. The cache database 136 may or may not have a corresponding index depending on its size and acceptable access speed. (See Fig. 2 and page 9, lines 14-22).

Fig. 5 describing the prior art and Fig. 6 describing the present invention illustrate database read procedures for systems operating without and with a key in the index entries, respectively. In the procedure illustrated by prior art Fig. 5, a client 130 transmits a query across the network 140 to the file server 100 to read data from the file server 100 (step 500). The query includes an identifier for the data row requested and a respective key. The file server 100 ultimately compares the key of a cached data row 235 with the key of a master data **row** 210 which has been requested (step 525). If the keys are the same, the file server 100 sends a reply to the client 130 that the cached data is still valid (step 540). If, however, the keys are different, the file server 100 transmits the entire data row 210 (or only the requested columns of the data row 210) to the client 130 to update the cache database 136 (step 535). The client 130 receives the response and acts accordingly (step 545). (See page 11, line 19 to page 12, line 4).

In the procedure of Fig. 6 illustrating the present invention, steps 600 to 620 are equivalent to steps 500 to 520 of prior art Fig. 5. However, in step 625, the file server 100 compares the key in the index entry of the requested data row 210 with the key of the cached data row 235 as received in the query. Thus, the key of an index entry is read and used in the comparison in step 625 of the present invention rather than the key of a row from the master database 126 as in prior art Fig. 5. If the keys are the same, the file server 100 transmits a message to the client 130 that the cached data is still valid (step 640). If, however, the keys are different, the file server 100 accesses the master database 126 to retrieve the current data row (step 630). The current data row is then returned to the client 130 to update the cache database 136 (step 635). (See page 12, lines 5-15).

The advantage of reading the key of an <u>index entry</u> and using it in the comparison in step 625 of the present invention (rather than the key of a <u>row</u> from the master database 126 as illustrated in step 525 of prior art Fig. 5) is that it avoids the need to read the whole row from the master database 126 and thus reduces the processing load on the file server 100. (See page 12, line 19 to page 13, line 6.)

Although the specification of the present application describes the use of time stamps as an example of the "keys" discussed above, other methods for marking data rows to enable data consistency comparisons are possible such as an incremental counter field or unique codes representative of the specific state of a row. (See page 13, lines 24-34.)

## <u>CONCISE EXPLANATION OF THE ISSUE PRESENTED FOR REVIEW</u>

Whether claims 1-9 are patentable under 35 U.S.C. §103 as not having been made "obvious" over Micka et al (U.S. Patent No. 5,592,618) in view of Antognini et al (U.S. Patent No. 5,649,185).[1]

[1] As noted above, the rejection of claims 1, 6 and 9 under 35 U.S.C. §101 has been withdrawn and therefore does not form an issue presented for review.

## WHETHER THE CLAIMS STAND OR FALL TOGETHER

Claims 1, 3 and 4 stand or fall together and do not stand or fall with any other claim(s).

Claims 2, 7 and 8 stand or fall together and do not stand or fall with any other claim(s).

Claim 5 stands or falls alone and does not stand or fall with any other claim(s).

Claim 6 stands or falls alone and does not stand or fall with any other claim(s).

Claim 9 stands or falls alone and does not stand or fall with any other claim(s).

## ARGUMENTS WITH RESPECT TO THE ISSUE PRESENTED FOR REVIEW

Claims 1-9 are patentable under 35 U.S.C. §103 as not having been made "obvious" over Micka et al (U.S. '618, hereinafter "Micka") in view of Antognini et al (U.S. '185, hereinafter "Antognini").

In order to establish a prima facie case of obviousness, all of the claimed limitations must be taught or suggested by the prior art. The combination of Micka and Antognini fails to teach or suggest all of the claimed limitations. For example, the combination fails to teach or suggest the following limitation of independent claim 1:

> "...comparing a first key stored in association with the item of data in the cache database with **a second key stored in association with an index entry** for the respective item of data in the master database (emphasis added)."

Similarly, the combination fails to teach or suggest the following limitations of independent claim 2:

> "reading a first key stored in association with a cached copy of a required item of data from the cache database;

> reading a **second key stored in association with an index entry** for a respective item of master data from the master database;
>
> comparing the first key with the second key...(emphasis added)."

Claims 1 and 2 each requires comparing a first key stored in associated with an item of data in a cache database with a second key stored in association with an **index entry** for a respective item of data in a master database.

The Rejection admits that Micka "does not disclose the step of comparing a first key stored in association with the item of data, in the cache database with a second key stored in association with an index entry for the respective item of data in the master database." (See page 3, lines 15-18 of the Rejection). Antognini fails to remedy this admitted deficiency of Micka.

Antognini discloses a data library system having a library catalog which associates each document with a single optimistic time stamp. (See col. 20, lines 11-12 of Antognini). Col. 20, lines 42-48 (specifically identified by the Office Action) of Antognini states "For the rare occasion when an application needs assurance that it has an 'up to the minute' copy, cache and catalog time stamps can be compared for the single document. This is helpful only if a communication echo is much faster than copying an image from the server to the client." However, this portion of Antognini merely discloses a conventional caching system in which a key (e.g., a time-stamp) associated with a data item in the cache is compared with a corresponding key in a master database. The key in the master database is stored in association with the data, not in association with an index of the data as required by independent claims 1 and 2 and their respective dependents. Antognini's "catalog time stamp", which can be compared to a "cache time stamp", provides a key in a master database (in this case, a library catalog) stored in association with the data (in this case, document data), rather than in association with an index of the data (document data).

The above portion of Antognini therefore appears to provide teachings similar to prior art Figs. 3b and 5 of the present application. As shown in prior art Figs. 3b and 5, a time stamp "TS" associated with a data row 210 of the master database (top row shown in Fig. 3b) is compared with a corresponding time stamp associated with a data item in the cache. In contrast, the present invention relates to comparing a key (e.g., a time stamp) associated with an index entry to a corresponding key of a data item in a cache. Fig. 3c of the application, illustrating an exemplary embodiment of the invention, shows a time stamp "TS" associated with the index 225 (bottom row shown in Fig. 3c), whereas no key is associated with index 225 (bottom row) of prior art Fig. 3b. By having the key associated with the index entry in the master database (rather than associated with the actual data of the master database as described in prior art Figs. 3b and 5 of the present application), the need to read the whole page containing a required data row to enable a comparison of the keys may be obviated. (See page 6, lines 3-8 and page 12, lines 27-35 of the application). Like prior art Figs. 3b and 5, Antognini fails to even appreciate these benefits of the present invention.

Since Antognini merely discloses a conventional caching system in which a key associated with a data item in the cache is compared with a corresponding key in a catalog database, the key in the catalog database being stored in association with the data rather than in association with the index entry of the data of the catalog database, Antognini fails to remedy the above admitted deficiencies of Micka. Accordingly, even if Antognini and Micka were combined as proposed by the Office Action, the combination would not have taught or suggested all of the claimed limitations.

Moreover, Appellant submits that one of ordinary skill in the art would not have been motivated to combine the teachings of Antognini's library catalog system with Micka's emergency disaster recovery system. The state table 700 illustrated in Fig. 4 (apparently identified by the Rejection as teaching the claimed index) and master journal 800 illustrated in Fig. 5 of Micka's emergency disaster recovery system enable secondary host 411 to store data received from primary host 401 in the correct order and enable secondary host 411 to operate on a stand-alone basis if primary host 401 ceases to exist. State table 700 is used and stored by secondary host 411 to enable secondary host 411 to know that it has successfully received all of the data transmitted to it and how and where to store the received data so that secondary host 411 can successfully and exactly replicate what is stored at primary host 401. State table 700 is not stored or used by primary host 401.

State table 700 used by secondary host 411 is not automatically updated once data on primary host 401 is updated since state table 700 is stored on secondary host 411, not primary host 401. Checking any item (any alleged key) associated with state table 700 would therefore not assist in checking if a data item (on the same database as the alleged index) was consistent to a corresponding data item stored on another database. Antognini's library catalog system is concerned with checking whether or not a data item in a cache is current compared to a corresponding data item in a catalog database. Since state table 700 of Micka would not be useful at all in establishing this consistency check, Appellant submits that there would have been no motivation to combine the teachings of Micka and Antognini.

The Rejection alleges "Specifically, referring to FIG. 4 of Micka, the reference discloses a mirroring process whereby an index is mirrored." However, as discussed above, state table 700 is used only by secondary host 411 (and not primary host 401) to help it know what it should have received from primary host 401 and where and how to store it. State table 700 therefore does not disclose an index for a respective data item in a **master** database or a key stored in association with the index entry for comparison of that key to another key stored in association with an item in a cache database.

Master journal 800 disclosed in Fig. 5 of Micka does not disclose the claimed index entry for a respective item of data in a master database, or a key stored in association with an index entry. Col. 11, lines 2-5 of Micka states, "The state table 700 and master journal 800 support disaster recovery, and hence must be able to operate in a stand-alone environment wherein the primary system 401 no longer exists." State table 700 and master journal 800 are thus not stored in the primary system 401. Indeed, col. 6, lines 30-31 of Micka states "FIG. 5 is a master journal as used by the secondary site of FIG. 1." State table 700 and/or master journal 800 therefore does not disclose an index for a respective data item in a **master** database or a key stored in association with the index entry for comparison of that key to another key stored in association with an item of data in a cache database.

Independent claim 2 further requires "retrieving in the event the first and second keys are the same the cached copy of the item of data or in the event the first and second keys are different the respective item of master data." Appellant respectfully submits that the combination of Micka and Antognini fails to further teach or suggest this additional feature. The Rejection alleges that col. 13, lines 28-36 of Micka teaches this feature. (See page 4, lines 17-19 of the Rejection). Appellant respectfully disagrees. Col. 13, lines 28-36 of Micka states the following:

> "If a time interval group is incomplete, then step 1110 retries reading the record sets from the primary storage controller 405 until the required data is received. If errors occur, a specific duplex volume pair or pairs may be failed. Having received complete time interval groups, step 1120 determines a first consistency group journal record. The first (or current) consistency group journal record is that record which contains the earliest time of update 610."

This portion of Micka discloses groups of data being transferred from primary host 401 to secondary host 411. Secondary host 411 checks whether it has received all of the expected records within a data group as determined from a state table. If a data group is incomplete, data from primary host 401 is reread so that it may be properly transferred from primary host 401 to secondary host 411. <u>This portion of Micka has nothing to do with retrieving data from either a cache database or a master database based on whether the compared first and second keys are the same or different</u>. If anything, this portion of Micka highlights the difference between the present invention which relates to caching and the Micka system which relates to disaster recovery (dr).

In the process in which secondary host 411 checks if a data group is complete and requests that data from primary host 401 be reread if the data is incomplete, no further action is taken with respect to the received data in the event that the data is complete. That is, the received data is not passed to some requesting party. Micka therefore clearly fails to teach or suggest "retrieving in the event the first and second key are the same the cached copy of the item of data."

In the normal operation of Micka's system, if a client wished to retrieve a data item, the client would retrieve it automatically from primary host 1200. Only if a disaster prevents the primary host 1200 from providing the data item would the client instead make a direct request to the secondary data host 1210 for the desired data item. The client would thus never make a single request expecting it to be fulfilled by either the primary or secondary data host 1200 or 1210 as determined automatically by the combination of the primary and secondary data hosts.

Page 4, lines 12-16 of the Rejection states the following:

> "reading a first key stored in association with a cached copy of a required item of data from the cache database (see Fig 7, step 1060, Micka);
>
> reading a second key stored in association with an index entry for a respective item of master data from the master database (see column 13, lines 2-8, Micka);
>
> comparing the first key with the second key (see Fig. 4, steps 710 and 711, Micka);"

The Rejection therefore appears to allege that the above identified portions of Micka (Fig. 7, step 1060, col. 13, ll. 2-8 and Fig. 4, steps 710-711) Micka disclose "reading a first key stored in association with a cache copy of a required item of data...", "reading a second key stored in association with an index entry for a respective item of master data..." and "comparing the first and second key...." This apparent allegation contradicts the earlier admission of the Rejection on page 3, lines 15-18 of the Rejection that "he [Micka] does not disclose the step of comparing a first key stored in association with the item of data, in the cache database with a second key stored in association with an index entry for the respective item of data in the master database." There is no detailed discussion whatsoever of Antognini with respect to claim 2 or claims 3-9.

Step 1060 in Fig. 7 involves communicating from primary host 401 journal records (i.e., a summary of the data it is about to transmit) to secondary host 411 so that secondary host 411 can determine if it correctly receives the full amount of data which the primary host 401 proceeds to send to the secondary host 411. Col. 13, lines 2-8 of Micka merely describes how the secondary host 411 uses previously received state table information to check whether it correctly receives all of the data which primary host 401 has indicated it will send. There is no teaching or suggestion in this portion of Micka of a master database, an index entry of that master database and/or a key stored in association with the index entry for a respective item in the master database. Items 710 and 711 are merely items in state table 700. Items 710 and 711 in Fig. 4 of Micka are corresponding entries in state table 700 stored in secondary host 411 in order that (a) secondary host 411 can ensure that it has correctly received all of the data which the primary host 401 intends to send; and (b) secondary host 411 can subsequently locate a requested piece of information from a secondary data store on the basis of information about the data as it was stored on the primary (in the event that the primary becomes unavailable as a result of a disaster).

Independent claim 5 requires a database file server that includes means for accessing an index of the database and reading an index entry for requested data items, the <u>index entry including a second key for the stored item of information</u>. Once again, Appellant submits that the combination of Micka and Antognini fails to teach or suggest a key associated with an index entry. The combination further fails to disclose comparing a first key for a previously retrieved copy of data with the second key from the index entry of the database. The apparent allegation on page 5, lines 5-10 in the Rejection that Micka discloses these features contradicts the Rejection's earlier admission that Micka does not disclose "comparing a first key stored in association with the item of data, in the cache database with a second key stored in association with an index entry for the respective item of data in the master database." (See page 3, lines 16-18 of the Office Action). There is no further detailed discussion of Antognini with respect to claim 5.

Claim 5 further requires means for returning an indication that the cached data item is consistent with the database if the two compared keys are the same or returning a copy of the requested item of data from the database if the two compared keys are different. There is no disclosure of this feature in Micka. The Final Rejection indicates that col. 10, lines 35-47 of Micka discloses this claimed feature. Appellant respectfully disagrees. Col. 10, lines 35-47 merely discloses determining whether all of the records in a primary host have been properly transferred to the secondary host. This portion of Micka has absolutely nothing to do with returning an indication of data consistency or returning a copy of requested data from a master database depending on whether two compared keys are the same or different.

The first part of the above-identified passage of Micka (lines 35-41) describes which parts of a "record set" (i.e., the summary of the data to be sent) are used by secondary host 411 to determine whether the data intended to be sent has actually been sent. The second part (lines 41-45) of the above-identified passage of Micka describes parts of the record set which are used to enable the secondary host 411 to correctly modify the data which it already has stored in order that it can correctly reflect the data as stored on the primary host at the time that the data was transmitted to the secondary host for backup. There is no teaching or suggestion of returning an indication that a previously sent item of data is still current or returning a requested item of data, and certainly no teaching or suggestion of returning an indication or returning a requested item of data based on the comparison of different keys.

Independent claim 6 is directed to a database index having an index entry which includes version information which changes each time respective information in a database changes. The Rejection apparently alleges that col. 11, lines 6-11 of Micka discloses this claimed feature. Appellant respectfully disagrees. Col. 11, lines 6-11 states the following:

> "A time stamp control is placed at the front and back of each master journal 800 to ensure that the entire control entry was successfully written. The time stamp control is further written to the secondary DASDs 417. The control elements include dual entries (1) and (2), wherein one entry is always a current entry, for example: (1) Timestamp control /Control Info/ Timestamp Control...."

While this portion of Micka discloses a time stamp being placed on a master journal entry, this portion of Micka fails to disclose changeable version information of a database index which changes each time information in the database itself changes. The time stamp in master journal 800 enables disaster recovery to take place using the most up to date, but still valid, set of data back-up by the secondary host in the event of a disaster. However, master journal 800 is not an index. An index stores locations of items of data such that a search can be performed through the index in order to determine where a requested item of data is located rather than having to search through the entire database looking for the requested item of data directly. (See page 9, lines 1-12 of the application). Master journal 800 does not provide a search mechanism to enable the location of a particular item of data to be located without having to search the entire database.

Col. 10, lines 49-55 (also identified in the Rejection) merely describes how state table 700 stores a mapping between locations on the primary host store and corresponding locations on the secondary host store. This portion of Micka does not describe an index.

Independent claim 9 requires "...comparing a first key stored in association with the item of data in the cache database and **a second key stored as a component of an index entry** for the respective item of data in the master database (emphasis added)." The Rejection admits that Micka does not disclose "the step of comparing a first key stored in association with the item of data, in the cache database with a second key stored in association with an index entry for the respective item of data in the master database." (See page 3, lines 15-18 of the Rejection.) Accordingly, Appellant respectfully submits that Micka fails to disclose the above-noted feature of claim 9. Antognini merely discloses a conventional caching system which involves the comparison of a key associated with a data item in the cache with a corresponding key for a stored data item in a catalog database (rather than stored in association with an index entry of the respective item of data in the catalog database) and thus fails to remedy the above deficiency of Micka.

In view of the foregoing, it is respectfully submitted that claims 1-9 are patentable as not having been made "obvious" over Micka in view of Antognini.

## CONCLUSION

For all of the reasons set forth above, it is respectfully requested that this appeal be granted and that the rejections discussed above be reversed.

Respectfully submitted,

**NIXON & VANDERHYE P.C.**

By: ____________________
Raymond Y. Mah
Reg. No. 41,426

RYM/sl
1100 North Glebe Road, 8th Floor
Arlington, VA 22201-4714
Telephone: (703)816-4000
Facsimile: (703)816-4100

## APPENDIX OF CLAIMS ON APPEAL

1. A method for checking the consistency of an item of data in a cache database with a respective item of data in a master database by comparing a first key stored in association with the item of data in the cache database with a second key stored in association with an index entry for the respective item of data in the master database.

2. A method for retrieving an item of data from one of a cache or a master database, the master database comprising a plurality of items of master data and an index containing entries corresponding to one or more of the items of master data, the cache database containing a cached copy of at least one item of the master data, the method comprising the steps of:

reading a first key stored in association with a cached copy of a required item of data from the cache database;

reading a second key stored in association with an index entry for a respective item of master data from the master database;

comparing the first key with the second key; and

retrieving in the event the first and second keys are the same the cached copy of the item of data or in the event the first and second keys are different the respective item of master data.

3. A method according to claim 1, wherein the first and second keys are time-stamps.

4. Use of a method according to claim 1 in a client/server system.

5. A database fileserver apparatus comprising:

input means for receiving a conditional read request for an item of data stored in the database, the request including a first key for a previously-retrieved copy of the item of data;

means for accessing an index of the database and reading an index entry for the requested item of data, the index entry including a second key for the stored item of information;

means for comparing the first and second keys; and

means if the keys are the same for returning an indication that the previously-retrieved copy of the item of data is consistent or if the keys are different for reading from the database and returning a copy of the item of data.

6. A database index, wherein at least one index entry in the index includes at least:

identity information for identifying an item of information in the database;

location information for indicating the location in the database of the item of information; and

version information which changes each time the respective information in the database changes.

7. A method according to claim 2, wherein the first and second keys are time stamps.

8. Use of method according to claim 2 in a client/server system.

9. A method for checking the consistency of an item of data in a cache database with a respective item of data in a master database by comparing a first key stored in association with the item of data in the cache database with a second key stored as a component of an index entry for the respective item of data in the master database.